Optics corrections in Booster

C.Y. Tan

2. April 2014

Abstract

Optics corrections in Booster was started with work by M. McAteer and A. Petrenko using LOCO implemented in a sets of scripts written in TCL, Mathematica, Matlab, octave and elegant. These scripts were later consolidated into a set of C++ programs that greatly improved the speed of finding the optics corrections. This report will look at the mathematics behind LOCO, describe in detail its C++ implementation, and show how to use the programs to correct Booster optics.

Table of Contents

1. Introduction

Booster optics correction was originally accomplished with a set of Mathematica, octave programs, elegant, TCL and ACL scripts written by M. McAteer and A. Petrenko [1]. After these physicists left Fermilab, the responsibility for optics corrections became the domain of physicists who were not familiar with the LOCO method. In order to get up to speed quickly, and to understand the process, the following goals were adopted:

1. Port all Mathematica programs to C++.
2. Make MADX the standard rather than elegant.
3. Increase the speed of obtaining results by 10 fold.
4. Create an integrated optics correction program that can run on the control system.

If these goals are met then Booster optics corrections will become a more push button operation and so any one can perform optics corrections without the involvement of experts. Of course, whether this ambitious goal can be realized and not forever be relegated to the realm of fantasy remains to be seen.

The method adopted by McAteer and Petrenko can be described in the following manner:

1. A set of orbit responses from the kicks of every corrector dipole of Booster is collected.
2. LOCO (discussed in section 2) is used to analyze the orbit responses from which the tilts and calibrations of the kickers and BPMs and the error quadrupole gradients can be derived.
3. These values are used to calculate the required skew quadrupole values to decouple Booster. A new set of orbit responses is collected again.
4. LOCO is applied to get a new set of tilts and calibrations of the kickers and BPMs and the error quadrupole gradients.
5. The corrections to the β beatings are the sum of the error quadrupole gradients found by LOCO, the pre-calculated corrector quadrupole settings that correct the effects of the dogleg (see section 3.2.2), and the corrector quadrupole settings that the user inputs for setting the required betatron tunes (see section 3.2.3).
6. These new corrector quadrupole currents are uploaded into Booster and a new set of orbit responses are collected.
7. Another set of orbit responses are collected again, and LOCO is used once more to check that the β beatings are indeed corrected.

In the following sections, the new implementation will be described. As always, the manual will probably be the laggard compared with the implementation and so it is best to contact those who are responsible for the latest incarnations.

2. LOCO

LOCO stands for *L*inear *O*ptics from *C*losed *O*rbits [2] and is used to calculate the quadrupole gradients, kicker strengths and tilts, BPM horizontal and vertical calibrations and tilts by minimizing the difference between the orbit response of the beam (defined in section 2.1) in Booster and the ideal Booster lattice model when it is kicked in the machine. Of course, since the goal is to not lose the beam from these kicks, it is clear that the kicks must be small so that the beam remains in its "closed orbit" but large enough so that the response remains linear and a good signal to noise ratio can be maintained.

In particular, the LOCO algorithm as applied to Booster will be discussed in this report. The way Booster is operated presents some interesting challenges to its implementation because it is a rapid cycling machine and radial feedback is used to keep the radial position of the beam fixed. The discussion in this section will be broken up into the following parts:

1. The general idea behind the LOCO algorithm when applied to Booster.
2. The C++ LOCO implementation.
3. LOCO results applied to decoupling Booster.
4. Use of MADX to correct beta beatings.
5. Correction of Booster optics with the results from MADX.

2.1 The LOCO algorithm applied to Booster

The principle of LOCO in Booster is the measurement of the orbit of the beam (i.e. orbit response) at every bpm as each dipole kicker in the ring is used to consecutively 1-bump the beam up the ramp at predefined breakpoints. Mathematically, the process can be parameterized as follows for each breakpoint *m*[1]

$$\left.\begin{aligned}\frac{\Delta x_i}{\Delta\theta_j} = {} & \frac{\sqrt{\beta_i\beta_j}}{2\sin\pi\nu_x}\cos\left(|\psi_i-\psi_j|-\pi\nu_x\right)- \\ & \frac{D_i}{D_{\mathrm{RPOS}}}\frac{\sqrt{\beta_{\mathrm{RPOS}}\beta_j}}{2\sin\pi\nu_x}\cos\left(|\psi_{\mathrm{RPOS}}-\psi_j|-\pi\nu_x\right)\end{aligned}\right\}\text{ at breakpoint } m \qquad (1)$$

where $\Delta\theta_j$ is the size of the dipole kick from the *j*th kicker, Δx_i is the change

[1] Everything from sections 2.1 to 2.3 is a summary of the algorithms implemented by M. McAteer and A. Petrenko in their programs for Booster.

in position at BPM i and $\beta_{i,j}$ are the beta functions at BPM i or kicker j respectively, and ν_x is the horizontal betatron tune. The RPOS device here is the radial feedback monitor. The beta function β_{RPOS} and dispersion D_{RPOS} at RPOS is required in the formula because Booster uses radial feedback and thus the dispersion function has a large effect on the response. The same formula *without* the dispersion term is used when the vertical plane orbit response is considered by appropriately substituting horizontal β's for vertical ones.

It is obvious that the same equation, Eq. 1, is found when the "measurement" comes from the lattice model.

In the subsequent formulas, the subscripts "measured" and "model" will be used to differentiate between the two. Clearly, both these methods will yield different values for $\Delta x_i / \Delta\theta_j$ and so a χ^2 error can be defined and it is

$$\chi^2 = \sum_{i,j} \left[\left(\frac{\Delta x_i}{\Delta\theta_j} \right)_{\mathrm{measured}} - \left(\frac{\Delta x_i}{\Delta\theta_j} \right)_{\mathrm{model}} \right]^2 \frac{1}{\sigma_{ij}^2} \tag{2}$$

where σ_{ij} is the statistical error taken at the 95% confidence interval of the slope in the linear fit of the measured orbit response at the *i*th BPM from a set of kicks using the *j*th kicker. The slope here is $(\Delta x_i / \Delta\theta_j)_{\mathrm{measured}}$ and the statistical error of the model is assumed to be the same as the measured error σ_{ij}. $\Delta x_i / \Delta\theta_j$ **is defined to be the orbit response of the beam at bpm *i* due to the kick from kicker *j*.** [2]

In principle, LOCO minimizes the χ^2 error, but in practice Eq. 2 is not directly used in the algorithm as the measure. An equivalent measure is defined instead so that a linear algebra problem can be constructed. See reference [3].

To this end, an error value ϵ_{ij} is defined and it is

$$\epsilon_{ij} = \frac{1}{\sigma_{ij}} \left[\left(\frac{\Delta x_i}{\Delta\theta_j} \right)_{\mathrm{measured}} - \left(\frac{\Delta x_i}{\Delta\theta_j} \right)_{\mathrm{model}} \right] \tag{3}$$

And then these error values are stacked to create an error vector $\boldsymbol{\epsilon}$ given by

$$\boldsymbol{\epsilon} = \begin{pmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \\ \vdots \\ \boldsymbol{\epsilon}_{N_{\mathrm{kicker}}} \end{pmatrix} \text{ where } \boldsymbol{\epsilon}_j = \begin{pmatrix} \epsilon_{1\mathrm{j}} \\ \epsilon_{2\mathrm{j}} \\ \vdots \\ \epsilon_{N_{\mathrm{bpm}} j} \end{pmatrix} \text{ error vector from kicker } j \tag{4}$$

where the number of bpms is N_{bpm} and the number of kickers is N_{kicker}. And thus when the norm of $|\boldsymbol{\epsilon}|$ is minimized, it is exactly equivalent to minimizing χ^2.

[2] This is the brilliant insight where slopes are used rather than absolute positions!

Now comes the interesting part – what to vary to mimimize $|\boldsymbol{\epsilon}|$. Let the parameters like kicker tilts, strengths, bpm tilts and calibrations and quadrupole strengths be the parameters that will be varied so that $|\boldsymbol{\epsilon}|$ is minimized. Let these parameters be called $\boldsymbol{K}$. It is clear that by making $(\Delta x_i / \Delta \theta_j)_{\text{model}}$ be functions of $\boldsymbol{K}$, and $(\Delta x_i / \Delta \theta_j)_{\text{measured}}$ be independent of $\boldsymbol{K}$, that the result of minimizing $|\boldsymbol{\epsilon}|$ is the discovery of the actual (in the hypothetical sense) tilts and calibrations of Booster. This is exactly what LOCO will endeavor to accomplish!

The process described above can be written mathematically as follows:

From the above discussion, $\boldsymbol{\epsilon}$ is a function of $\boldsymbol{K}$ by construction, i.e. $\boldsymbol{\epsilon}(\boldsymbol{K})$. Then for small variations in $\boldsymbol{K} \rightarrow \boldsymbol{K}_0 + \Delta \boldsymbol{K}$, $\boldsymbol{\epsilon}(\boldsymbol{K})$ can be Taylor expanded about $\Delta \boldsymbol{K}$ [3]

$$\boldsymbol{\epsilon}(\boldsymbol{K}) = \boldsymbol{\epsilon}(\boldsymbol{K}_0 + \Delta \boldsymbol{K}) \approx \boldsymbol{\epsilon}(\boldsymbol{K}_0) + \left.\frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}}\right|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \boldsymbol{K} \qquad (5)$$

where $\partial \boldsymbol{\epsilon}(\boldsymbol{K}) / \partial \boldsymbol{K}$ is the Jacobian. Minimization of χ^2 essentially means that in the ideal case $\boldsymbol{\epsilon}(\boldsymbol{K}) = 0$, i.e.

$$\boldsymbol{\epsilon}(\boldsymbol{K}_0) + \left.\frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}}\right|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \boldsymbol{K} = 0 \;\Rightarrow\; \left.\frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}}\right|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \boldsymbol{K} = -\boldsymbol{\epsilon}(\boldsymbol{K}_0) \qquad (6)$$

Therefore, by inverting $\partial \boldsymbol{\epsilon}(\boldsymbol{K}) / \partial \boldsymbol{K}$, $\Delta \boldsymbol{K}$ is solved and some solution for the tilts and calibrations of the Booster are found. However, $\partial \boldsymbol{\epsilon}(\boldsymbol{K}) / \partial \boldsymbol{K}$ is not square because there are more equations than unknowns, i.e. the problem is over-constrained. There are several ways to invert $\partial \boldsymbol{\epsilon}(\boldsymbol{K}) / \partial \boldsymbol{K}$ but SVD is the method chosen in LOCO. Again, it must be emphasized that the answer is not unique and it is given by

$$\Delta \boldsymbol{K} = -\left(\left.\frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}}\right|_{\boldsymbol{K}=\boldsymbol{K}_0} \right)^{-1} \boldsymbol{\epsilon}(\boldsymbol{K}_0) \qquad (7)$$

2.1.1 Obtaining the Jacobian

Eq. 7 is in principle easy to solve once $\partial \boldsymbol{\epsilon}(\boldsymbol{K}) / \partial \boldsymbol{K}$ is known. However, the problem is that analytic formulæ do not exist for it. Therefore, an approximation must be used. It is easy to see that Eq. 5 can be used to find $\partial \boldsymbol{\epsilon}(\boldsymbol{K}) / \partial \boldsymbol{K}$ because

$$\left.\frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}}\right|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \boldsymbol{K}_u = \boldsymbol{\epsilon}(\boldsymbol{K}_0 + \Delta \boldsymbol{K}_u) - \boldsymbol{\epsilon}(\boldsymbol{K}_0) \qquad (8)$$

where a new variable $\Delta \boldsymbol{K}_u$ has been introduced that signifies a small user

[3] Reminder: Both $\boldsymbol{K}$ and $\Delta \boldsymbol{K}$ must be dimensionless.

selected change in the calibrations or tilt of the model. Now, here comes the clever part, let $\Delta \boldsymbol{K} = \Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}} \boldsymbol{\alpha}$, which is defined below:

$$\Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}} \boldsymbol{\alpha} = \begin{pmatrix} \Delta K_{u_1} & 0 & \cdots & 0 \\ 0 & \Delta K_{u_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \Delta K_{u_{N_{\text{parameters}}}} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{N_{\text{parameters}}} \end{pmatrix} = \Delta \boldsymbol{K} \tag{9}$$

where the diagonal entries of the matrix $\Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}}$ are the elements of the vector $\Delta \boldsymbol{K}_{\boldsymbol{u}}$. Substituting this into Eq. 6, it becomes

$$\left(\left. \frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}} \right|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}} \right) \boldsymbol{\alpha} = -\boldsymbol{\epsilon}(\boldsymbol{K}_0) \tag{10}$$

Now, if the parameter changes $\Delta \boldsymbol{K}$ are varied one at a time, i.e.

$$\Delta \boldsymbol{K}_{\boldsymbol{u}_1} = \begin{pmatrix} \Delta K_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ \Delta \boldsymbol{K}_{\boldsymbol{u}_2} = \begin{pmatrix} 0 \\ \Delta K_2 \\ \vdots \\ 0 \end{pmatrix}, \ \ldots, \ \Delta \boldsymbol{K}_{\boldsymbol{u}_{N_{\text{parameters}}}} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \Delta K_{N_{\text{parameters}}} \end{pmatrix} \tag{11}$$

then when only $\Delta \boldsymbol{K}_{\boldsymbol{u}_k}$ is applied,

$$\begin{aligned} \left. \frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}} \right|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \boldsymbol{K}_{\boldsymbol{u}_k} &= \left. \frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}} \right|_{\boldsymbol{K}=\boldsymbol{K}_0} \begin{pmatrix} 0 \\ \vdots \\ \Delta K_k \\ \vdots \\ 0 \end{pmatrix} \\ &= \Delta K_k \times \left(k \text{ th column of } \left. \frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}} \right|_{\boldsymbol{K}=\boldsymbol{K}_0} \right) \end{aligned} \tag{12}$$

Thus, the rhs of Eq. 12 is just Eq. 5 when $\Delta \boldsymbol{K} = \Delta \boldsymbol{K}_{\boldsymbol{u}_i}$, i.e.

$$\boldsymbol{\epsilon}(\boldsymbol{K}_0 + \Delta \boldsymbol{K}_{\boldsymbol{u}_k}) - \boldsymbol{\epsilon}(\boldsymbol{K}_0) = \Delta K_k \times \left(k \text{ th column of } \left. \frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}} \right|_{\boldsymbol{K}=\boldsymbol{K}_0} \right) \tag{13}$$

Therefore, when the above is substituted into Eq. 9, the result is

$$\begin{aligned} \left. \frac{\partial \boldsymbol{\epsilon}(\boldsymbol{K})}{\partial \boldsymbol{K}} \right|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}} &= \Big(\boldsymbol{\epsilon}(\boldsymbol{K}_0 + \Delta \boldsymbol{K}_{\boldsymbol{u}_1}) - \boldsymbol{\epsilon}(\boldsymbol{K}_0) \quad \boldsymbol{\epsilon}(\boldsymbol{K}_0 + \Delta \boldsymbol{K}_{\boldsymbol{u}_2}) - \boldsymbol{\epsilon}(\boldsymbol{K}_0) \quad \ldots \\ &\qquad \ldots \ \boldsymbol{\epsilon}(\boldsymbol{K}_0 + \Delta \boldsymbol{K}_{\boldsymbol{u}_{N_{\text{parameters}}}}) - \boldsymbol{\epsilon}(\boldsymbol{K}_0) \Big) \end{aligned} \tag{14}$$

which means that the vector $\boldsymbol{\epsilon}(\boldsymbol{K}_0 + \Delta \boldsymbol{K}_{\boldsymbol{u}_k}) - \boldsymbol{\epsilon}(\boldsymbol{K}_0)$ is the *k*th column of the matrix $\partial \boldsymbol{\epsilon}(\boldsymbol{K})/\partial \boldsymbol{K}|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}}$.

Notice that from Eq. 3, the kth column of the matrix is the vector $\partial \boldsymbol{\epsilon}(\boldsymbol{K})/\partial \boldsymbol{K}|_{\boldsymbol{K}=\boldsymbol{K}_0} \Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}}$ that has elements that come from the response of the ith bpm from the jth kicker

$$\begin{aligned}[\boldsymbol{\epsilon}(\boldsymbol{K}_0+\Delta \boldsymbol{K}_{\boldsymbol{u}_k})-\boldsymbol{\epsilon}(\boldsymbol{K}_0)]_{ij} &= -\frac{1}{\sigma_{ij}}\left[\left(\frac{\Delta x_i}{\Delta \theta_j}\right)_{\text{model}}\Bigg|_{\boldsymbol{K}=\boldsymbol{K}_0+\boldsymbol{K}_{\boldsymbol{u}_k}} - \left(\frac{\Delta x_i}{\Delta \theta_j}\right)_{\text{model}}\Bigg|_{\boldsymbol{K}=\boldsymbol{K}_0}\right] \\ &\equiv -\frac{\Delta \boldsymbol{x}_i}{\Delta \boldsymbol{\theta}_j}\Bigg|_{\Delta \boldsymbol{K}_{\boldsymbol{u}_k}}\end{aligned} \tag{15}$$

where the last line introduces a new variable that economizes the subsequent notation. Hence, Eq. 10 becomes

$$\left(\frac{\Delta \boldsymbol{x}}{\Delta \boldsymbol{\theta}}\Bigg|_{\Delta \boldsymbol{K}_{\boldsymbol{u}_1}} \quad \frac{\Delta \boldsymbol{x}}{\Delta \boldsymbol{\theta}}\Bigg|_{\Delta \boldsymbol{K}_{\boldsymbol{u}_2}} \quad \cdots \quad \frac{\Delta \boldsymbol{x}}{\Delta \boldsymbol{\theta}}\Bigg|_{\Delta \boldsymbol{K}_{\boldsymbol{u}_{N_{\text{parameters}}}}}\right)\boldsymbol{\alpha}=+\boldsymbol{\epsilon}(\boldsymbol{K}_0) \tag{16}$$

where the kth column of the matrix on the lhs is the vector whose entries contain the response of each bpm from each kick when parameter $\Delta \boldsymbol{K}_{\boldsymbol{u}_k}$ is changed. This matrix is inverted using SVD to solve for $\boldsymbol{\alpha}$. Once $\boldsymbol{\alpha}$ is found, $\Delta \boldsymbol{K}$ can be calculated using Eq. 9.

Further refinements of $\boldsymbol{\alpha}$ can be obtained by iteration, i.e by letting $\Delta \boldsymbol{K}_{\boldsymbol{u}} \leftarrow \Delta \hat{\boldsymbol{K}}_{\boldsymbol{u}} \boldsymbol{\alpha}$ and solving for $\boldsymbol{\alpha}$ again using Eq. 16. Of course, one can only hope that $\Delta \boldsymbol{K}$ (i.e. $\boldsymbol{\alpha} \rightarrow \boldsymbol{0}$) converges to a value that is close to reality!

2.1.1.1 Typical size of the Booster Jacobian

Notice that $\partial \boldsymbol{\epsilon}(\boldsymbol{K})/\partial \boldsymbol{K}$ is an $(N_{\text{bpms}} \times N_{\text{kickers}}) \times N_{\text{parameters}}$ matrix because the number of tilts and calibrations of kickers, bpms and quads for a typical Booster LOCO calculation is $N_{\text{parameters}}=442$ and the number of orbit responses is $(N_{\text{bpms}} \times N_{\text{kickers}})=9700$ and thus $(N_{\text{bpms}} \times N_{\text{kickers}}) \times N_{\text{parameters}}=4287400$. This is clearly a large memory problem!

2.1.2 Visualizing the goodness of fit

The best way to ascertain whether the calibrations, rolls and tilts found by LOCO are good is to somehow compare them to the measurements and then cleverly displaying them so that any problems can be recognized immediately.

There are two types of plots that can give the notion of the goodness of fit:

1. The trend of the parameter α as a function of the number of refinements. If the fits are getting better $\alpha \to 0$ for every parameter $\boldsymbol{K}$. Of course, α contains the measurements, because it is buried in its derivation from Eq. 3.
2. A 3-D plot with $\epsilon(\boldsymbol{K}_0)$ plotted against each BPM i on the x-axis and kicker j on the y-axis. Any BPM or kicker problems should stick out like a sore thumb. This 3-D plot will have to be repeated for every break point.

2.1.2.1 Example

The visualization of the goodness of fit using data taken on 15 Nov 2013 is shown here. The first pass of LOCO produces some the quality control fits discussed above. For example, the convergence of the α parameter for BPM tilts is shown in Fig. 2.1. It is clear from this figure that there is one bpm that is behaving in a strange manner and it is identified as BPML18. Similarly, when $|\epsilon(\boldsymbol{K}_0)|$ is plotted again each BPM and kicker, the problem BPMs are easily found in the plots shown in in Fig. 2.2.

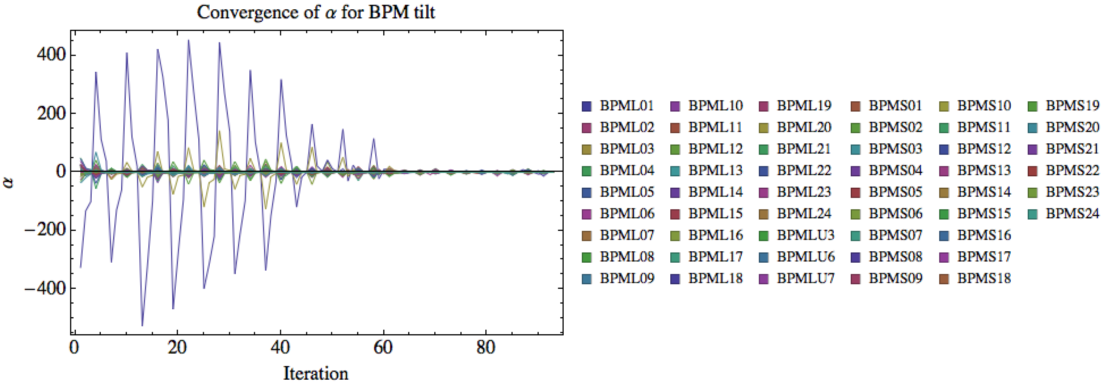

Figure 2.1: *This is an example of how the α parameter converges for the tilt of 51 bpms. There is a clear problem for one of them and it is identified to be BPML18.*

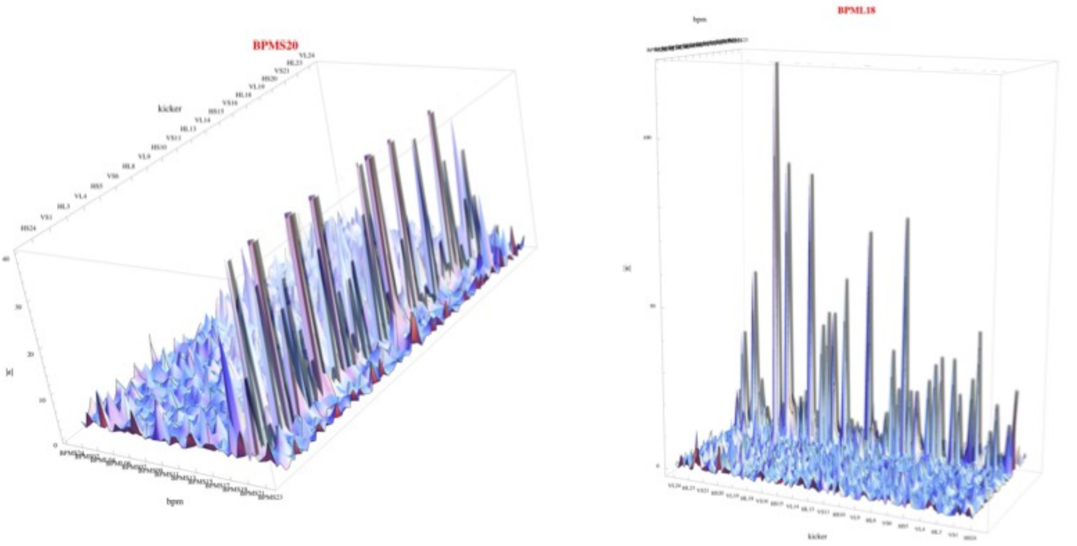

Figure 2.2: *This is an example of* $|\epsilon(K_0)|$ *taken at 3.1 ms into the ramp plotted against each BPM and kicker. It is obvious that there are problems with BPMS20 and BPML18.*

2.2 Booster model

The Booster magnets are combined function magnets and so the quadrupole components are already incorporated into these magnets and scale according to the dipole current. Therefore, in the ideal Universe, the magnets have been designed to give a perfect periodic lattice without any corrector quadrupole, skew quadrupole or sextupole magnets. See Fig. 2.3. Unfortunately, the dogleg at L3 is a source of large optical distortions because of the significant edge focusing of these magnets. These magnets run DC and their effect on the beam diminishes as its energy increases.

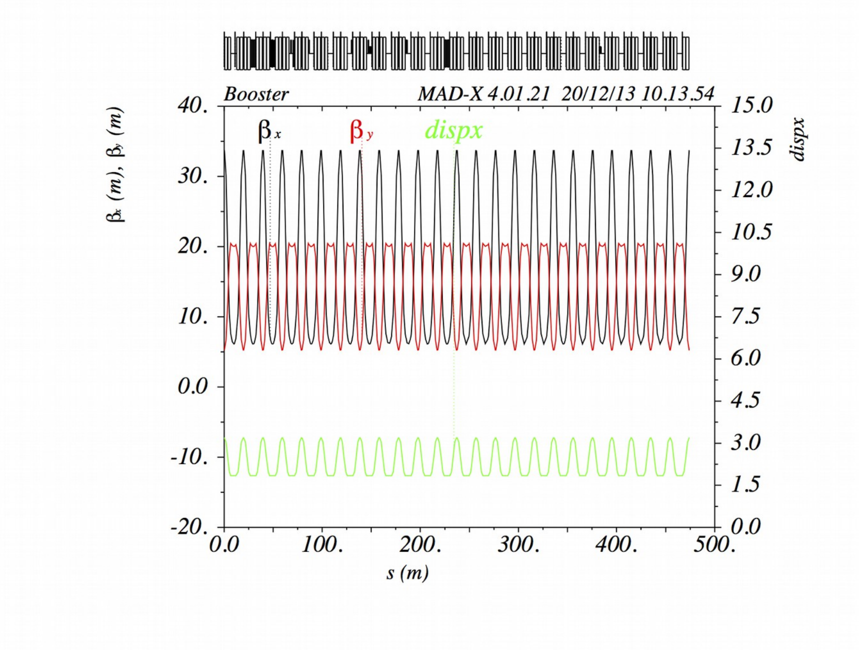

Figure 2.3: *The Booster lattice when all the pseudo and corrector quadrupoles (QL, QS, SQL and SQS; Q1err, Qserr, SQLerr, SQSerr) and ORBUMP have zero current. The combined function magnets provide the focusing and defocusing required for a periodic lattice. The dogleg is off in this plot. The results when the dogleg is added is shown shown in Fig. 3.2.*

Besides the dogleg, if there are errors in the combined function magnets, or any other sources of error, the tunes and coupling have to be corrected. And in Booster, there are corrector quadrupole and skew quadrupole magnets [4] that are used to correct tunes and coupling. These set of corrector quadrupoles are called QL1 - 24 and QS1 - 24; and the corrector skew quadrupoles are called SQL1 - 24 and SQS1 - 24.

In the Booster MADX model, instead of directly using these corrector magnets to fix coupling and β beating, a set of pseudo quadrupoles are defined in the model. These are placed at the same position as the corrector quadrupoles and are called QLerr1 - Qlerr24 and QSerr1 - QSerr24; SQLerr1 - SQLerr24 and SQSerr1 - SQSerr24. LOCO varies these pseudo quadrupoles to fit the model to the measurement. The reason why pseudo quadrupoles are used will be made clear in section 3.

2.3 LOCO algorithm as implemented in fitit

The general program flow of fitit as pseudocode is shown below: (In order to map the C++ variables used in the implementation to the theory discussed in section 2.1, the theoretical variables are in [.]). This was translated from the original scripts written by M. McAteer and A. Petrenko.

```
1. Read in the orbit response data from *ORM_H.TXT and *ORM_V.TXT.
    i. For each kicker, read in the $N$ changes in kick strength for each break point on the ramp.
        a. For each bpm, calculate the slope and the 95% confidence interval of the slope of its response to the $N$ kicks. The slope is dx/dkick or $[(\Delta x_i/\Delta\theta_j)_{\text{measured}}]$.
        b. Store dx/dkick and the confidence interval into an orbit response structure called ormresp.
2. Read in the dispersion measurement data from *DISP.TXT for each break point on the ramp.
    i. Read in the $N$ changes in RPOS
        a. For each bpm, calculate the slope and the 95% confidence interval of the slope of its response to the $N$ RPOS changes. The slope is dx/drpos.
        b. Store dx/drpos and the confidence interval into a dispersion response structure called dresp.
3. Read in the parameters that will be fitted or varied from the params2vary.sdds file. It contains dipole corrector strengths, bpm calibrations and tilts, quad error strengths that will be varied. $[\boldsymbol{K}]$
4. Read in the initial set up of the Booster given in machine_parameters.sdds. $[\boldsymbol{K}_0]$. These are the initial settings of the variables and not the actual lattice settings.
5. For every break point (i.e. cycle time), fitit
    i. create a new optics file for MADX or elegant from the magnet currents at that break point from the MagnetSettings.txt file.
    ii. Calculate the Jacobian for the response of the beam to each dipole
```

kick

a. use MADX or elegant to calculate a theoretical orbit response given the strengths from Magnetsettings.txt at this break point.
    1. For MADX, an R-matrix is constructed from the result.

b. From the R-matrix, the response of the beam for each dipole kick can be calculated, i.e. what was done in the experiment and the d*/dkick can be calculated from R. Store the results of d*/dkick for each kicker as seen by every BPM in kpbmdata.

c. Apply tilts and calibrations previously read in from machine_params.sdds to kbpmdata.

d. Create a y0-vector that contains dx/dkick, dy/dkick, dx/drpos, dy/drpos and stack them in a predefined order. This vector is the reference. $[(\Delta x_i/\Delta\theta_j)_{\text{model}}]_{\boldsymbol{K}=\boldsymbol{K}_0}$

e. For each dipole corrector do
    1. change its calibration by dvalue.
    2. Apply the tilts and calibrations to kbpmdata with this new dvalue.
    3. Create a y1-vector that contains dx/dkick, dy/dkick, dx/drpos, dy/drpos and stack them in the correct order. $[(\Delta x_i/\Delta\theta_j)_{\text{model}}]_{\boldsymbol{K}=\boldsymbol{K}_0+\Delta\boldsymbol{K}}$
    4. Calculate the difference yerr from y1 – y0. $[\Delta\boldsymbol{x}/\Delta\boldsymbol{\theta}]$

f. For each dipole corrector do
    1. Change its tilt by dvalue.
    2. Apply the tilts and calibrations to kbpmdata with this new dvalue.
    3. Create a y1-vector that contains dx/dkick, dy/dkick, dx/drpos, dy/drpos and stack them in the correct order.
    4. Calculate the difference yerr from y1 – y0. $[\Delta\boldsymbol{x}/\Delta\boldsymbol{\theta}]$

g. For each bpm do
    1. Change its *x* gain by dvalue.
    2. Apply the tilts and calibrations to kbpmdata with this new dvalue.
    3. Create a y1-vector that contains dx/dkick, dy/dkick, dx/drpos, dy/drpos and stack them in the correct order.
    4. Calculate the difference yerr from y1 – y0. $[\Delta\boldsymbol{x}/\Delta\boldsymbol{\theta}]$

h. For each bpm do
   1. Change its *y* gain by dvalue.
   2. Apply the tilts and calibrations to kbpmdata with this new dvalue.
   3. Create a y1-vector that contains dx/dkick, dy/dkick, dx/drpos, dy/drpos and stack them in the correct order.
   4. Calculate the difference yerr from y1 – y0. $[\Delta \boldsymbol{x}/\Delta \boldsymbol{\theta}]$

i. For each bpm do
   1. Change its tilt by dvalue.
   2. Apply the tilts and calibrations to kbpmdata with this new dvalue.
   3. Create a y1-vector that contains dx/dkick, dy/dkick, dx/drpos, dy/drpos and stack them in the correct order.
   4. Calculate the difference yerr from y1 – y0. $[\Delta \boldsymbol{x}/\Delta \boldsymbol{\theta}]$

j. For each normal and skew quad do
   1. Change its strength by dvalue.
   2. Create a new optics file that contains this quad change.
   3. Use MADX or elegant to calculate orbit response. Store BPM results from each dipole kick in kbpmdata.
   4. Apply tilts and calibrations previously read in from machine_params.sdds to kbpmdata.
   5. Create a y1-vector that contains dx/dkick, dy/dkick, dx/drpos, dy/drpos and stack them in the correct order.
   6. Calculate the difference yerr from y1 – y0. $[\Delta \boldsymbol{x}/\Delta \boldsymbol{\theta}]$

k. Do three fit iterations on yerr to generate new tilts, calibrations and quad strengths for the dipoles, bpms and quads.
   1. use MADX or elegant to calculate a reference orbit response given the magnet strengths from the kbpmdata.
   2. Apply the present tilts and calibrations to kbpmdata.
   3. Create a reference y0 vector. Again y0 is the reference from the model. $[(\Delta x_i/\Delta \theta_j)_{\text{model}}]_{\boldsymbol{K}=\boldsymbol{K}_0}$
   4. Create a y vector from the measured orbit response and dispersion data, i.e. from ormresp and dresp that was read in earlier. $[(\Delta x_i/\Delta \theta_j)_{\text{measured}}]_{\boldsymbol{K}=\boldsymbol{K}_0}$
   5. Create y – y0. $[\boldsymbol{\epsilon}(\boldsymbol{K}_0)]$

6. Create M whose columns are yerr. $[\Delta x / \Delta\theta]$
7. Solve M^{-1}(y-y0) = dX using SVD to invert M. See Eq. 16.
8. Create a new set of dvalues from dX**,** dvalues → dX*dvalues $[\Delta K_u \leftarrow \Delta \hat{K}_u \alpha]$ and create a new machine_parameters.sdds file that consists of $K_0 + \Delta K_u$. This will be the new Booster "setup" at this break point for the next iteration.

l. Using the new calibrations and tilts, the following files twiss_vs_t.twi, machine_params_vs_t.sdds and optics_vs_t.sdds files are generated.

m. Continue iterating until all the break points are done.

3. Correcting Booster optics

This section deals with correcting Booster optics once the ΔK 's are found with LOCO. The first part is to decouple of Booster and the second part is to reduce the β beatings.

3.1 Decoupling

Once the pseudo skew quadrupole (SQLerr, SQSerr) errors strengths Δk_s are found by LOCO, these values can be directly used to decouple Booster.[4] The reasoning is as follows:

1. LOCO has found Δk_s for every pseudo skew quadrupole that makes the model look like the as found Booster. These pseudo skew quadrupole errors come from an unknown source and are assumed to be <u>independent</u> of the corrector skew quadrupole values k_s .
2. The as found Booster has corrector skew quadrupole (SQL, SQS) strengths k_s that are not zero and thus the total skew quadrupole corrector strength is $k_s + \Delta k_s \neq 0$ in the model. This means that the Booster model is globally coupled.
3. Thus, to decouple the Booster model, the real Booster skew quadrupole strengths must be set to $-\Delta k_s = k'_s$, so that when these new values (SQL, SQS) are read into the Booster model $k_s \rightarrow k'_s$, this makes $k'_s + \Delta k_s = (-\Delta k_s) + \Delta k_s = 0$ and so the Booster model is fully decoupled.
4. If the Δk_s 's found by LOCO is correct, then Booster is also decoupled.

Notice that this procedure can be iterative. Suppose the coupling is smaller but

[4] Algorithm supplied by M. McAteer.

still not zero even after $-\Delta \boldsymbol{k}_s$ has been loaded into Booster. LOCO can be applied again. If the new pseudo skew quadrupole value found by LOCO is $\Delta \boldsymbol{k}'_s$, then the new corrector skew quadrupole value that has to be loaded into Booster is $-\Delta \boldsymbol{k}'_s$. And *ad infinitum*.

3.1.1 Decoupling algorithm as implemented in **decouple**

The general program flow of decouple as pseudocode is shown below:

```
1. Read in the skew quadrupole magnet currents up the ramp from
   MagnetSettings.txt.
   a. Create the break points list.
2. Read in the pseudo skew quadrupole strengths from
   machine_params_vs_t.sdds.
   a. Convert skew quadrupole strengths to current.
3. For each break point do
   a. Calculate the new skew quadrupole currents for each break point. See
      section 3.1 for the formula.
4. Rescale the new skew quadrupole currents so that they are within the
   range of the power supplies.
5. Write out new skew quadrupole currents to file.
```

3.2 Correcting β beatings

The pseudo quadrupole (QLerr, QSerr) error strengths $\Delta \boldsymbol{k}_Q$ found by LOCO can be used to correct partially the β beatings in Booster. It is only partial because of the way these pseudo corrections found by LOCO are applied:

- The pseudo quadrupole values are used to <u>zero out all</u> quadrupole sums (QL – QLerr), (QS – QSerr) in the Booster model. See section 3.2.1.

In principle, doing this would reproduce the ideal periodic lattice shown in Fig. 2.3. However, when the dogleg is turned on, the model contains the effect from the edge focusing of the dogleg magnets in L3, and the resulting β errors propagate around the ring. See Fig. 3.2. And because (QLerr, QSerr) are found by fitting to this distorted lattice, the ideal lattice has to be recovered by hand. Therefore, after all the quadrupole settings (QL – QLerr), (QS – QSerr) are set to zero in the model, a small set of correction quadrupoles around the dogleg in long 3 have been selected to reduce the β beatings and to control the horizontal dispersion in the model. This is discussed in section 3.2.2.

Although not strictly related to the correction of β beatings, the betatron tunes will move after the corrections are applied and so part of the β beatings correction for Booster will include the mitigation of this effect.

Hence, this section is divided into three parts:

1. The use of $\Delta \boldsymbol{k}_Q$ found by LOCO for zeroing out the pseudo and quadrupole settings in Booster and its application to the real Booster.
2. Correction of the β errors caused by the edge effects of the dogleg.
3. Correction of the betatron tune.

3.2.1 Corrections with pseudo quadrupoles found by LOCO

In this part, the pseudo quadrupole error strengths found by LOCO are applied as follows:

1. LOCO has found $\Delta \boldsymbol{k}_Q$ for every pseudo quadrupole that makes the model look like the as found Booster. These pseudo quadrupole errors come from an unknown source and are assumed to be independent of the corrector quadrupole values $\boldsymbol{k}_Q$.
2. The as found Booster has corrector quadrupole (QL, QS) strengths $\boldsymbol{k}_Q$ that are not zero and thus the total quadrupole corrector strength is $\boldsymbol{k}_Q + \Delta \boldsymbol{k}_Q \neq 0$ in the model. This means that Booster has β errors. See Fig. 3.1
3. Thus, to fix the β errors in the Booster model, the real Booster corrector quadrupole strengths must be set to $-\Delta \boldsymbol{k}_Q = \boldsymbol{k}'_Q$, so that when these new values (QL, QS) are read into the Booster model $\boldsymbol{k}_Q \rightarrow \boldsymbol{k}'_Q$, this makes $\boldsymbol{k}'_Q + \Delta \boldsymbol{k}_Q = (-\Delta \boldsymbol{k}_Q) + \Delta \boldsymbol{k}_Q = 0$ and so the Booster model is fully β corrected.
4. If the $\Delta \boldsymbol{k}_Q$'s found by LOCO is correct, then the real Booster is β corrected *if* the dogleg is off. Section 3.2.2 will deal with the effects of the dogleg.

Note that this procedure only works if the combined function magnets in the actual Booster are perfectly modeled (see Fig. 2.3) and all the unknown sources of error (or even known sources of error that are in the installed combined function magnets) are captured in the pseudo and corrector quadrupoles. This procedure is quite different from machines that use separated function magnets because the $\Delta \boldsymbol{k}$'s are used to correct the quadrupole strengths by using the formula $\Delta I / I = -\Delta k / k$, where I is the current of the quadrupole that gives gradient k and ΔI is required the change in current that corresponds to the Δk found by LOCO. See for example Ref. [3].

Next, notice that this procedure can be iterative. Suppose β beating is smaller but still not zero even after $-\Delta \boldsymbol{k}_Q$ has been loaded into Booster. LOCO can be applied again. If the new pseudo quadrupole values found by LOCO is $\Delta \boldsymbol{k}'_Q$, then the new corrector quadrupole value that has to be loaded into Booster is $-\Delta \boldsymbol{k}'_Q$. And *ad infinitum*.

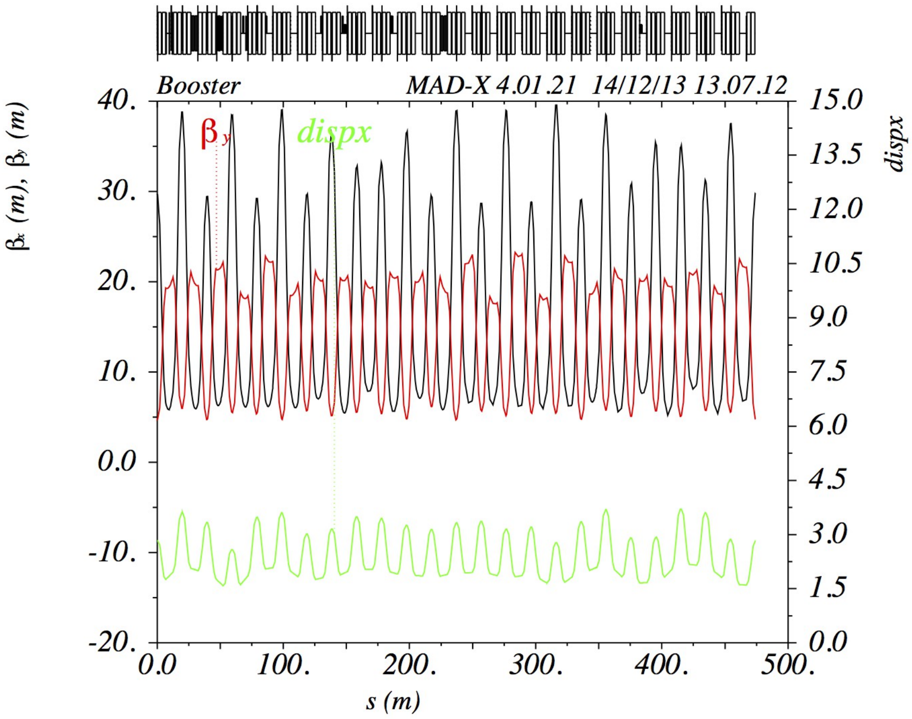

Figure 3.1: *The non zero values of the pseudo and corrector quadrupoles (QL, QS; QLerr, QSerr) and zero ORBUMP at 3.1 ms gives a non-periodic lattice compared to the ideal lattice shown in Fig. 2.3.*

It is expected that when the pseudo and corrector quadrupole settings are set to zero, the lattice would look like that shown in Fig. 2.3. It does indeed. But when the dogleg is turned on, it introduces β beatings into the model. See Fig. 3.2.

3.2.2 Correcting the effects of the dogleg[5]

The ideal lattice shown in Fig. 2.3 Becomes distorted when the dogleg is turned on. The distorted lattice shown in Fig. 3.2 is **the lattice that LOCO will fit to in the model.** What this means is that the LOCO results will give quadrupole

[5] The original larger set of quadrupoles and the scaling shown in Eq. 17 were found by V. Lebedev.

strengths that will never give the ideal lattice. Therefore, the β beatings that come from the effects of the dogleg (ORBUMP is zero after 2.05 ms) will need to be corrected by hand selecting a small subset of the available quadrupoles that are centered around the dogleg in long 3. They are:

1. QS24, QS01, QS02, QS03, QS04, QS05, QS06

No QL corrector quadrupoles are used because the effects of the dogleg are purely horizontal and the Booster model is fully decoupled.

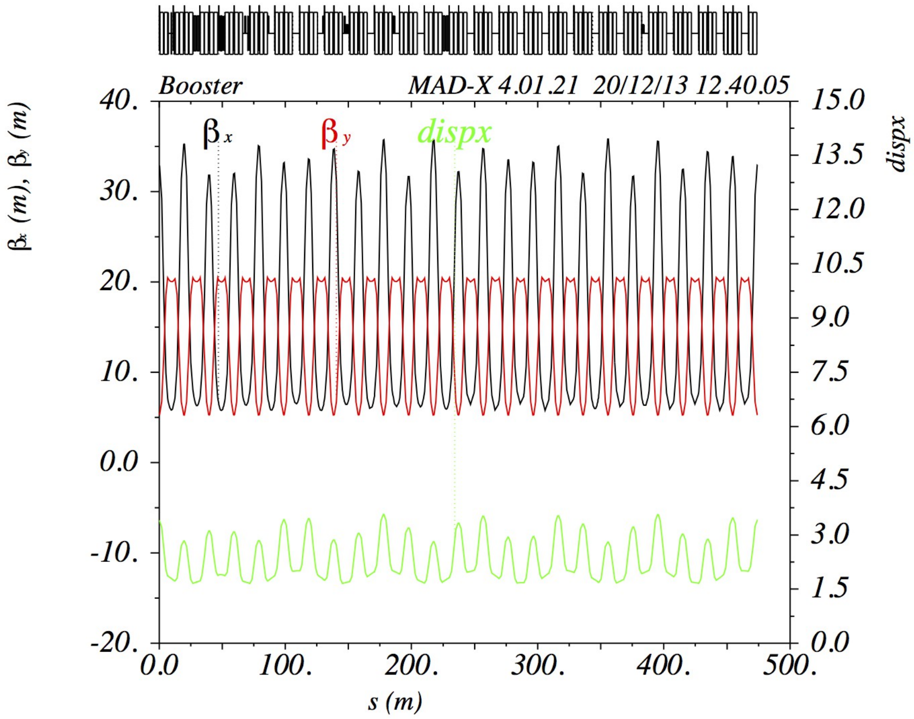

Figure 3.2: *This is the model lattice with the dogleg on but with all the pseudo and corrector quadrupoles (QL, QS, SQL and SQS; Qlerr, Qserr, SQLerr, SQSerr) and ORMUMP set to zero current. It is clear that the lattice in the horizontal plane no longer looks like the ideal one shown in Fig. 2.3.*

These quadrupoles are tuned to minimize the β beatings after the pseudo and quadrupole values have been set to zero and the dogleg has been set to 293.4 A. The gradient of these quadrupoles at injection are summarized in Table 3.1.

The gradient k_q of these quadrupoles at each break point scales as follows:

$$k_q(t) = k_{\mathrm{inj}}\, p_{\mathrm{inj}} / p(t) \qquad (17)$$

where $p_{\mathrm{inj}} = 954.25$ MeV/c is the momentum of the protons at injection, $p(t)$ is the momentum of the beam at time t of the ramp.

The scaling is inversely proportional to momentum because as the beam energy increases, the effect of the dogleg decreases because it does not ramp. The reason for this scaling is evident from the formula for the deflection angle $\Delta\theta=\int \boldsymbol{B}\cdot d\boldsymbol{l}/(p/c)$. If the dogleg setting at injection changes from 293.4 A, the gradients shown in Table 3.1 must be tweaked accordingly.

QS quadrupoles	k_{inj} (T/m)
QS24	0.403011
QS01	0.298202
QS02	0.185324
QS03	0.290666
QS04	0.372706
QS05	0.4282
QS06	–0.0412226

Table 3.1: *Selected quadrupoles and their gradients used for correcting β errors at injection arising from the dogleg setting of 293.4 A at the beam kinitic energy of 400 MeV.*

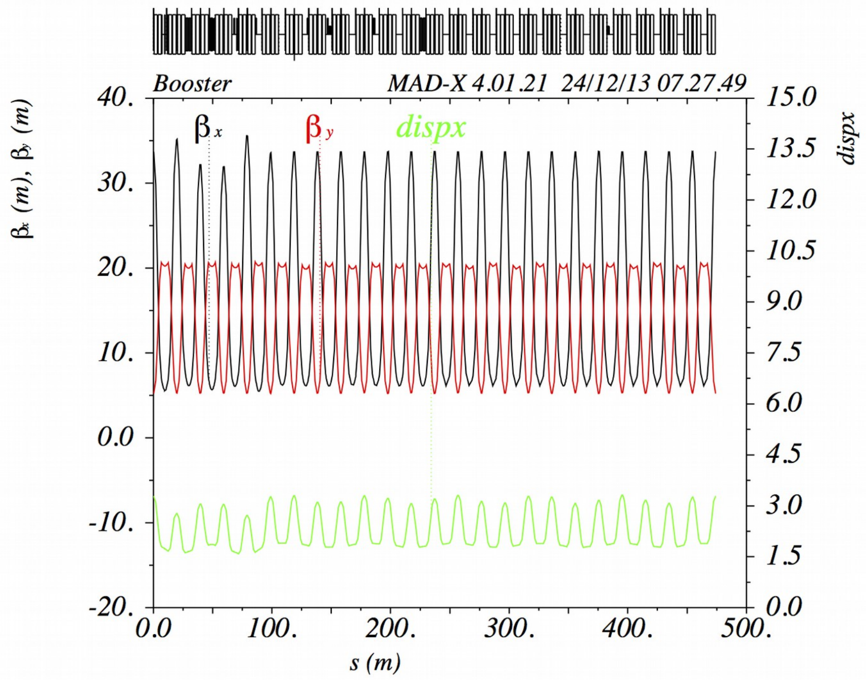

Figure 3.3: *By powering the quadrupoles shown in Table 3.1 with the values calculated using Eq. 17 at t= 3.1 ms (beam kinetic energy is 415 MeV), the β beating is reduced outside the dogleg region.*

The correction at 3.1 ms of the ramp (corresponds to the beam kinetic energy of 415 MeV) is shown in Fig. 3.3. Compared to Fig. 3.2, the β beating outside the dogleg region is clearly fixed while the β beating inside long 3 is clearly not fixed. At higher energies, the effect of the dogleg is greatly diminished. This can be seen in Fig. 3.4 for the beam kinetic energy of 7.0 GeV at 27.7 ms of the ramp.

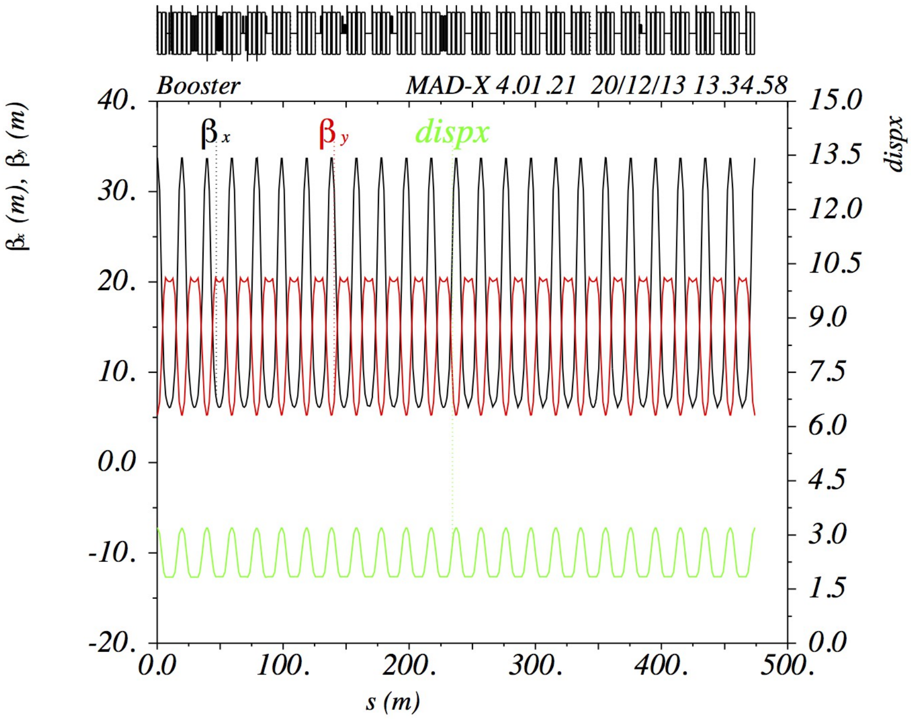

Figure 3.4: *At 27.7 ms (beam kinetic energy = 7 GeV), the effect of the dogleg is greatly diminished and the quadrupole strengths shown in Table 3.1 are lowered commensurately in strength using Eq. 17.*

3.2.3 Adding in as found quad error strengths

The as found global corrector quadrupole strengths are averaged and can be added to the new corrector quadrupoles, i.e. the average is average is calculated

$$\begin{aligned} \langle \mathrm{QL} \rangle &= \frac{1}{24}\sum_{i=1}^{24} \mathrm{QL}_i \\ \langle \mathrm{QS} \rangle &= \frac{1}{24}\sum_{i=1}^{24} \mathrm{QS}_i \end{aligned} \tag{18}$$

and these averages are added to the new QL and QS values found in section 3.2.1. How these values are used for corrections are shown in the section 3.2.4.

3.2.4 β beating correction algorithm as implemented in fixlattice

The general program flow of fixlattice as pseudocode is shown below:

```
1. Read in the quadrupole magnet currents up the ramp from
   MagnetSettings.txt.
    a. Create the break points list.
2. Read in the pseudo quadrupole strengths from machine_params_vs_t.sdds.
    a. Convert quadrupole strengths to current.
3. For each break point do
    a. Calculate the new quadrupole currents for each break point. See
       section 3.2 for the formula.
4. If enabled, add in the average of the QL and QS values (See Eq. 18) that
   are already in the Booster to each break point.
5. Rescale the new quadrupole currents so that they are within the range of
   the power supplies.
6. Write out new quadrupole currents to file.
```

4. Example

An example of a successful correction of the β beating is discussed here. It is important that the desired orbits in Booster have been established *before* any optics corrections are contemplated. The reason is that the error quadrupole corrections can be large in the first pass and if the beam is not centered in them, they can introduce a collection of 1-bumps that will affect the corrections found by LOCO.

4.1 As found Booster optics

The as found Booster optics that was measured on 15 Nov 2013 is for injection, at transition and extraction are shown in Fig. 4.1. It is clear that the as found optics is no where near the ideal. Furthermore, the optics collected at transition does not look believable. It is speculated that the short bunch length at this breakpoint causes problems for the BPMs and thus the measurement is not good.

4.2 First corrections

The as found Booster optics was measured on the 15 Nov 2013. The results are shown in Fig. 4.1. The as found betatron tunes are shown in Fig. 4.2.

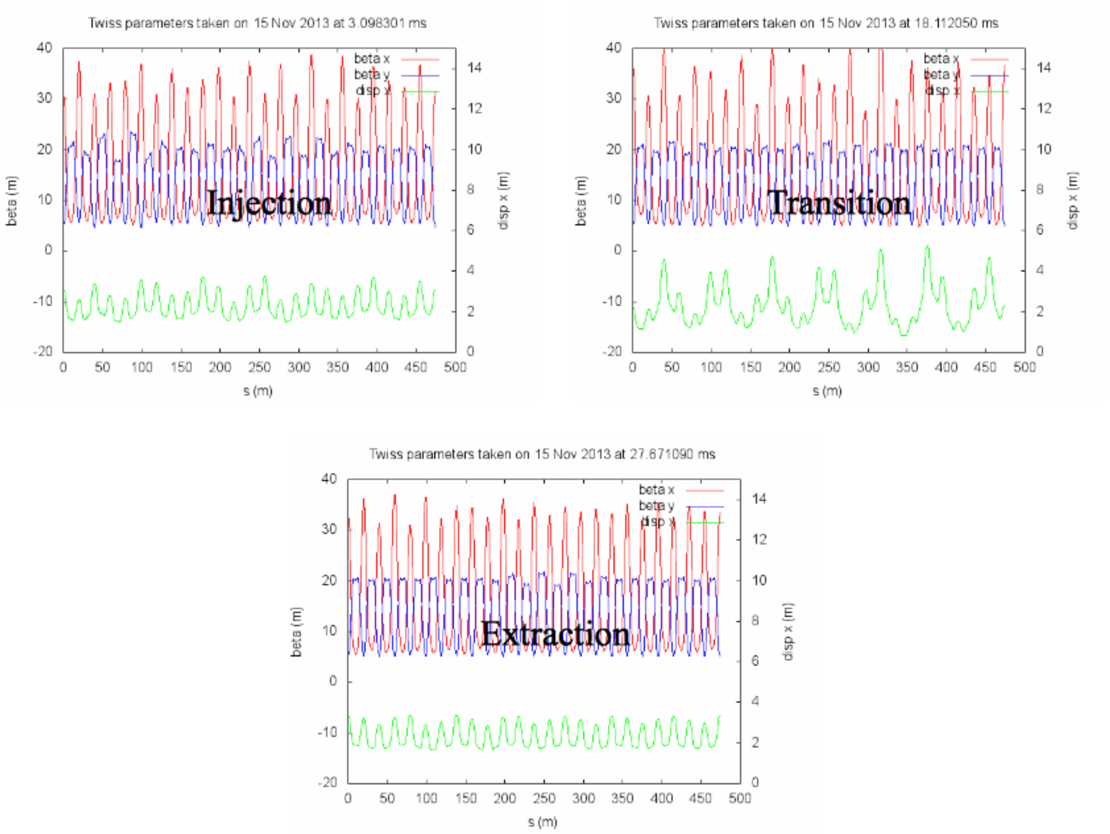

Figure 4.1: *The as found Booster lattice at injection, transition and extraction. The transition lattice cannot be trusted because the short bunch length causes problems for the BPMs.*

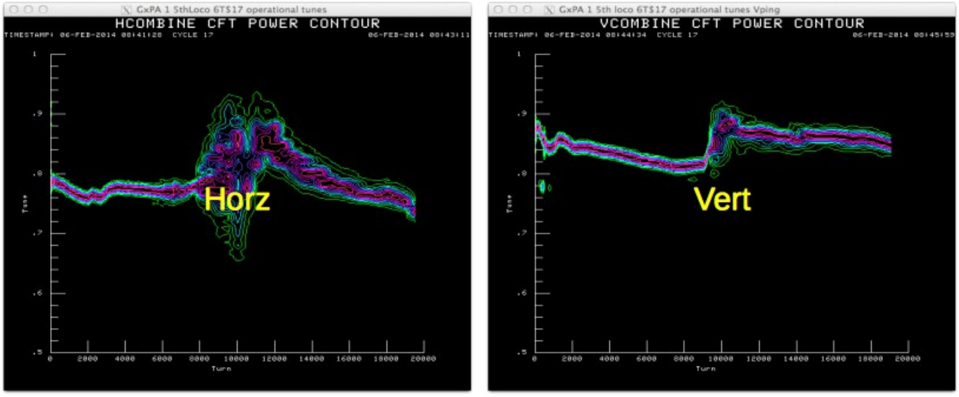
GxPA 1 5thLoco 6T$17 operational tunes
HCOMBINE CFT POWER CONTOUR
TIMESTAMP: 06-FEB-2014 08:41:28 CYCLE 17
06-FEB-2014 08:43:11
Horz
GxPA 1 5th loco 6T$17 operational tunes Vping
VCOMBINE CFT POWER CONTOUR
TIMESTAMP: 06-FEB-2014 08:44:34 CYCLE 17
06-FEB-2014 08:45:59
Vert
Tune
Turn

Figure 4.2: *The as found betatron tunes.*

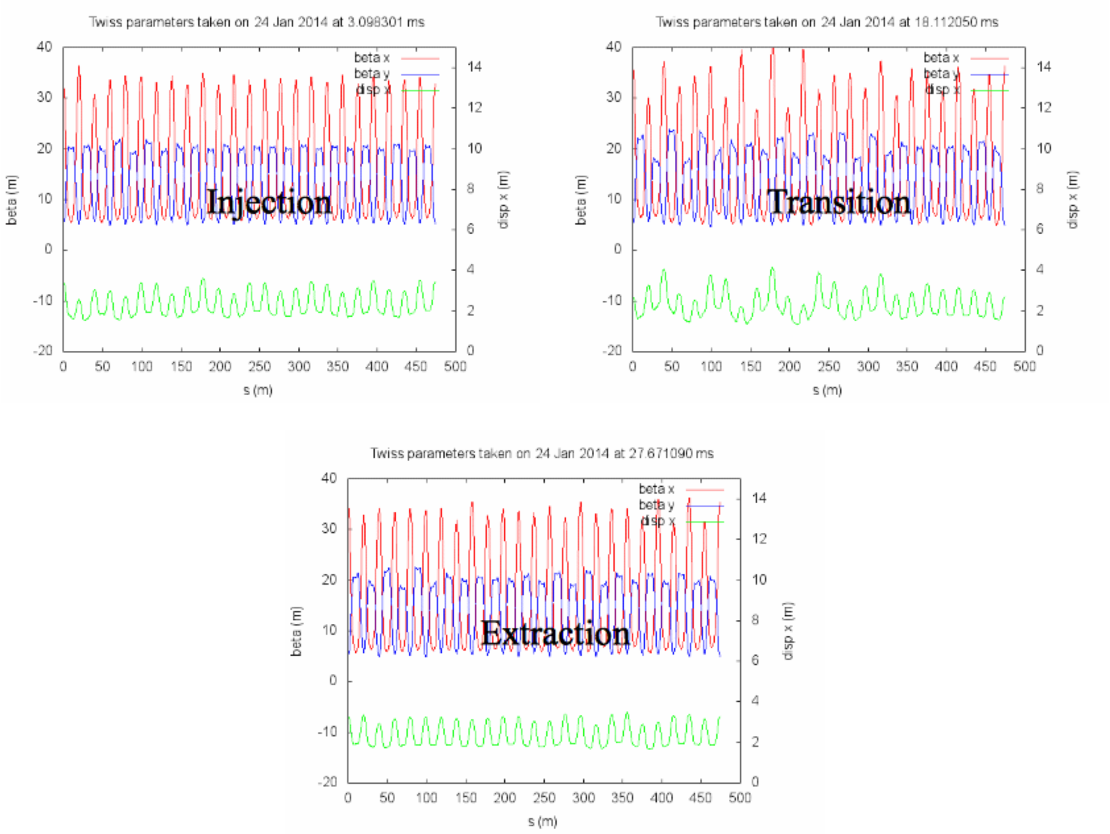

Figure 4.3: *These are the measured lattices after the corrections found on 15 Nov 2013 were loaded.*

The first set of corrections found by LOCO after the Booster was decoupled was calculated and loaded into Booster on 27 Jan 2014. Fig. 4.3 shows the results after the corrections found on 15 Nov 2013 were loaded in. It is clear from this figure that the corrections at injection reduces the beta beating while at extraction the corrections have a minimal effect.

However, on closer examination of the results, it was found that the corrections were effective before transition and had minimal effect after it. Furthermore, it was also discovered that the tunes after transition moved very far away from the required tunes for extraction which caused unacceptable losses. See Fig. 4.4. The source of the excursion were the large changes in the QL and QS currents near extraction. The currents in the QL and QS ramps before and after correction are shown in Fig. 4.5.

And because the corrections do not do much after transition, it was decided that only optics corrections before transition will be loaded into Booster. The new QL and QS ramps are shown in Fig. 4.6. The betatron tunes shown in Fig. 4.7, as expected, remain the same after transition. However, the tunes below transition are no longer where they are supposed to be and tune changes are needed to keep the losses low.

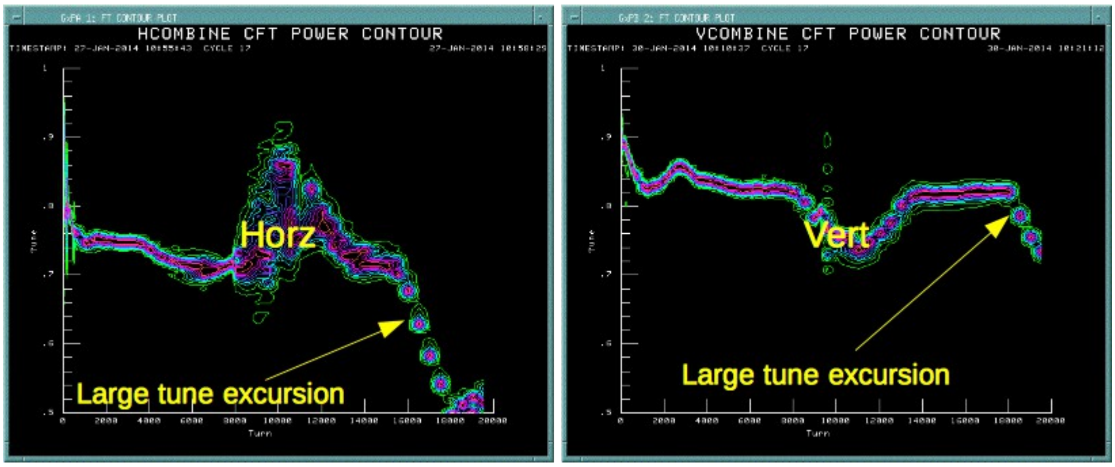

Figure 4.4: *After the corrections were loaded in QL and QS, there were large tune excursions after transition in both the horizontal and vertical planes. These tunes can be compared to the as found tunes shown in Fig. 4.2.*

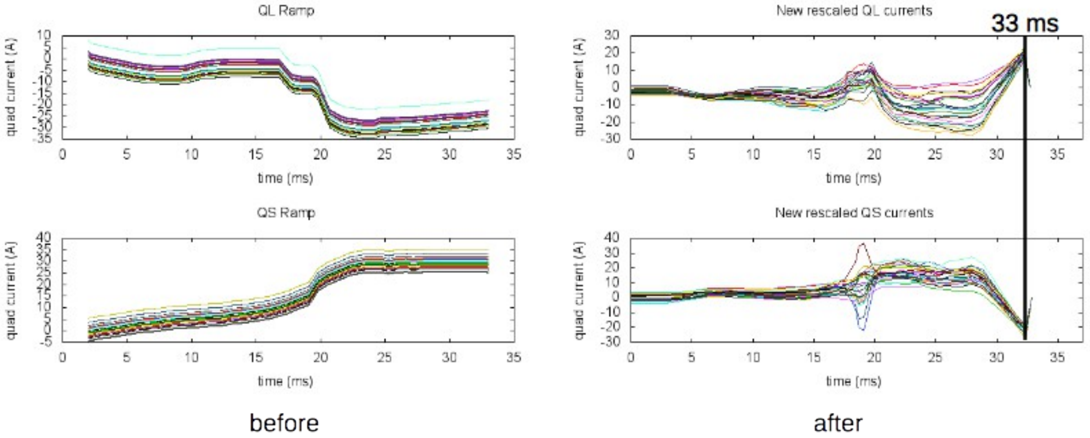

Figure 4.5: *These figures show that the source of the excursions in both planes come from the large changes in QL and QS after correction.*

For completeness, the lattice at injection, transition and extraction are shown in Fig. 4.8 when only the optics below transition are corrected. Comparing this figure with Figs. 4.1 and 4.3, it is easy to see that, indeed, only the injection lattice has been corrected.

After the below transition corrections are loaded, the betatron tunes below transition are not at the right place and losses are high. More tuning that does not change the lattice is required to fix this. Work is being done as of 24 Mar 2014 to get the 473 cards ready to take these new tune ramps.

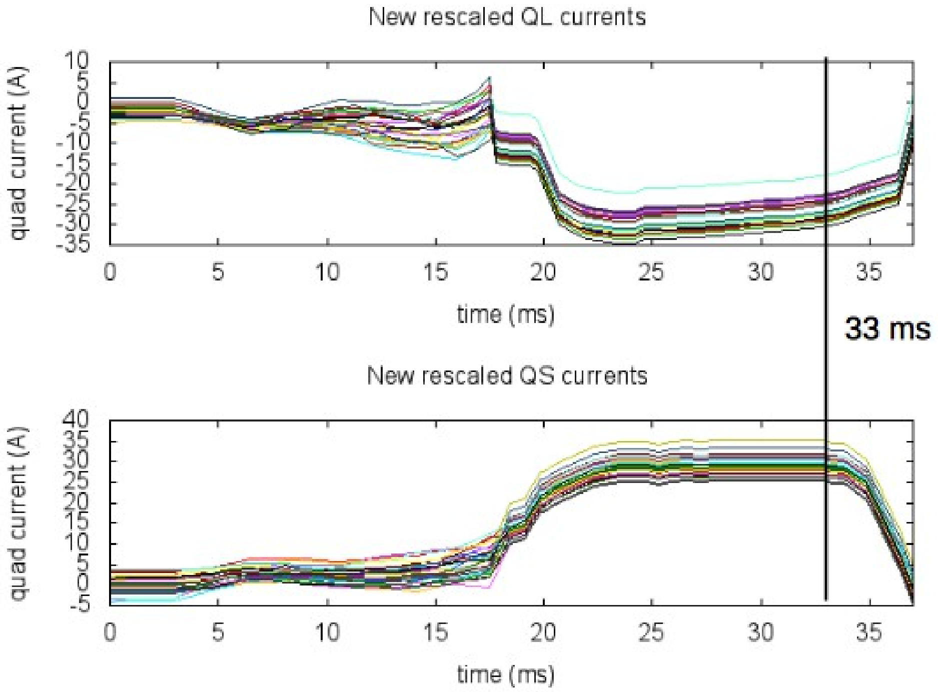

Figure 4.6: *QL and QS are are kept the same above transition and are only corrected below transition.*

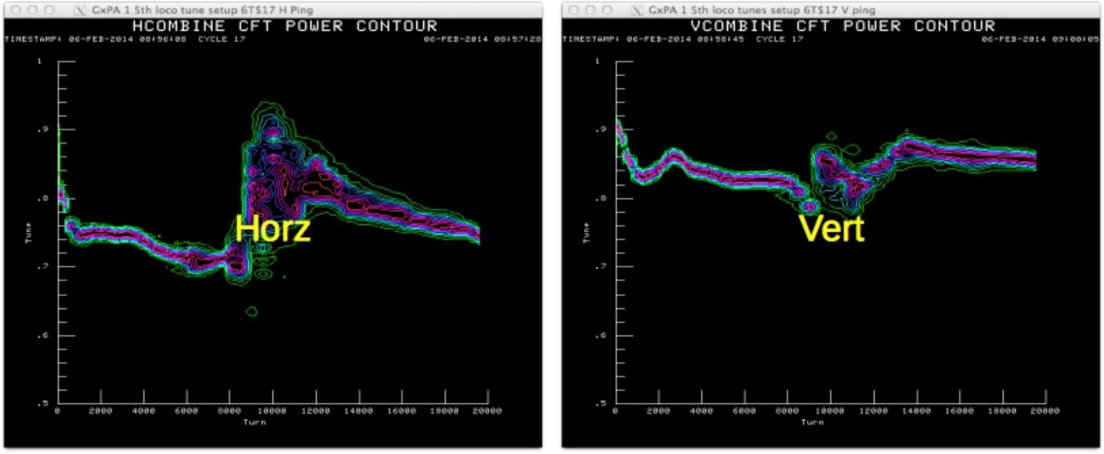
GxPA 1 5th loco tune setup 6T$17 H Ping
HCOMBINE CFT POWER CONTOUR
TIMESTAMP: 06-FEB-2014 08:56:08 CYCLE 17
06-FEB-2014 08:57:28
Horz
Tune
Turn
GxPA 1 5th loco tunes setup 6T$17 V ping
VCOMBINE CFT POWER CONTOUR
TIMESTAMP: 06-FEB-2014 08:58:45 CYCLE 17
06-FEB-2014 09:00:05
Vert
Tune
Turn
1
.9
.8
.7
.6
.5
0
2000
4000
6000
8000
10000
12000
14000
16000
18000
20000

Figure 4.7: *These are the betatron tunes after correction QL and QS below transition. It is obvious that the tune excursion above transition that was seen in Fig. 4.4 is no longer there, and that these tunes are the same as the as found tunes shown in Fig. 4.2 above transition.*

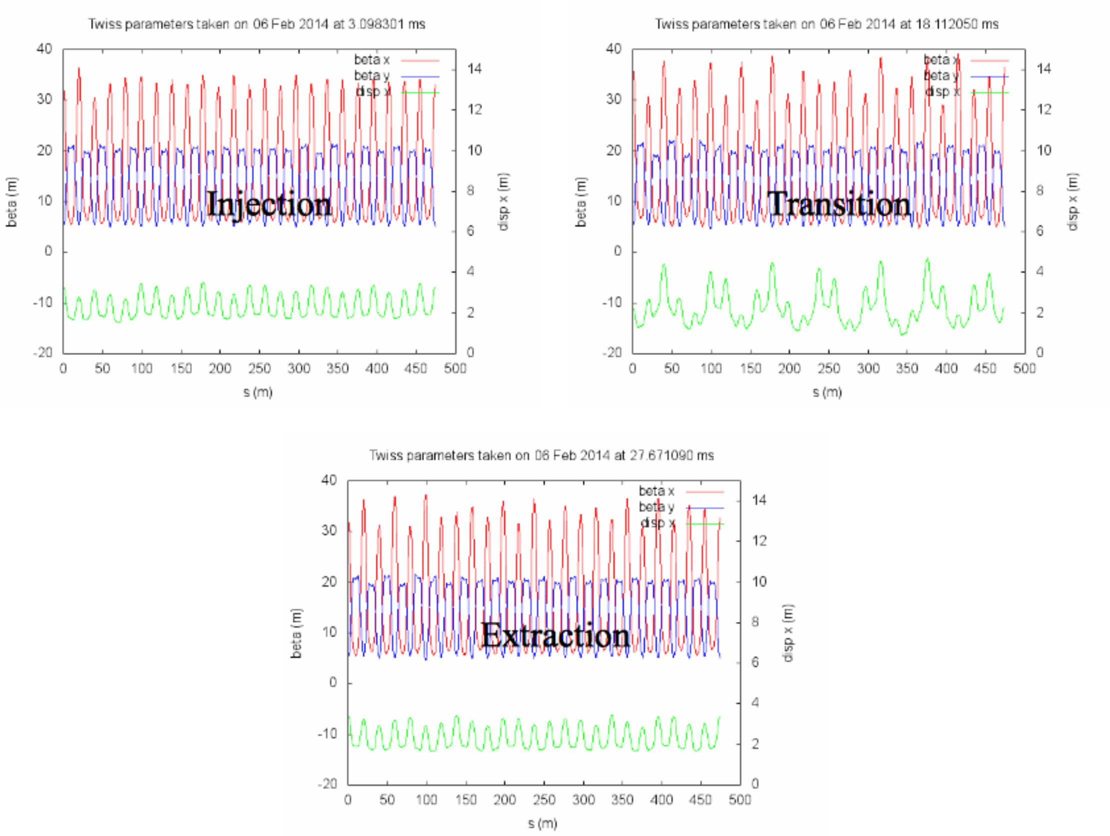

Figure 4.8: *This figure shows the optics corrections below transition only. The results should be compared to those shown in Figs. 4.1* and *4.3.*

5. Conclusion

Booster LOCO in C++ has been fully implemented and tested. Not all goals that were mentioned in the Introduction have been met yet:

1. Port all Mathematica programs to C++. **Done!**
2. Make MADX the standard rather than elegant. **Done!**
3. Increase the speed of obtaining results by 10 fold. **6× speed up!**
4. Create an integrated optics correction program that can run on the control system. **To be done.**

The results show that the method does work, but care must be taken to check the tunes. At this time 02 Apr 2014, the wait is for the 465 cards to be completely connected to the 473 cards that control the quadrupole ramps. Once this is done, the tunes can be easily corrected with B15.

6. Acknowledgments

The author would like to thank M. McAteer and A. Petrenko without whom LOCO for Booster and the current implementation would not have been possible. And V. Lebedev for his method for correcting the dogleg errors.

Appendix

LOCO is obviously parallelizable because in the construction of the Jacobian shown in Eq. 16, each column $\left(\Delta \mathbf{x}/\Delta \boldsymbol{\theta}\right)\big|_{\Delta \boldsymbol{K}_{u_j}}$ is generated for every dipole strength, and tilt, bpm x gain, y gain, and tilt, and quadrupole strength – $N_{\text{parameters}}$ in all – and these parameters are assumed to be independent. Therefore, the calculation of the columns of the Jacobian can be divided up among N processors. After each processor completes its calculation, the results are collected and placed into the columns of the Jacobian. The block diagram of the algorithm is shown in Fig. a.

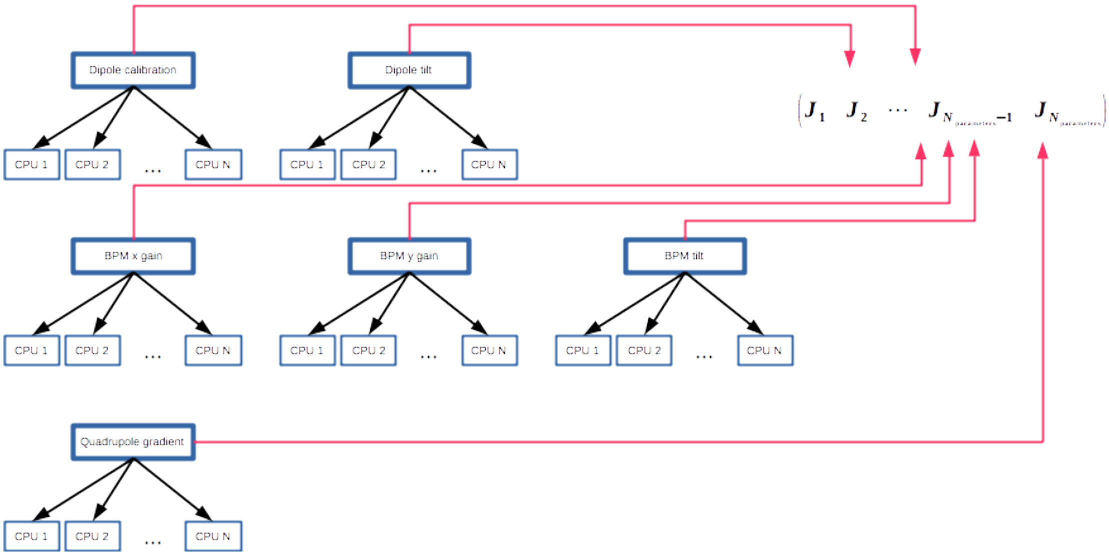

Figure a: *The problem is divided up between N processors for each parameter that needs to be calculated. The results are collected and stuffed into the columns of the Jacobian.*

After the Jacobian is created, it is inverted using SVD. At this time, SVD is only run on one processor and thus its completion time is independent of the number of CPUs used in parallel processing.

The results of using the parallel algorithm for one break point are shown in Figs. 2 and 3. The speed up is clearly non-linear and tops out at about 30 CPUs. The parallel algorithm is about 6× faster than the serial algorithm. The bottleneck is the SVD algorithm which can be parallelized if ScaLAPACK is used instead of LAPACK. This may be used in the future.

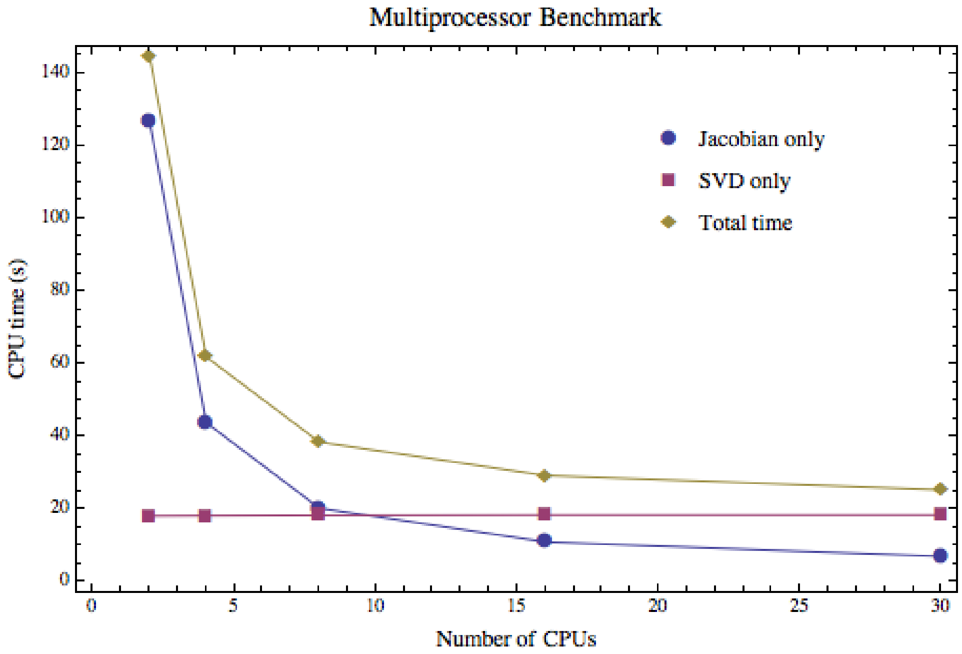

Figure 2: *The multiprocessor benchmarks show that as the number of CPUs is increased, the faster it takes to complete the calculation of one break point. However, the scaling is not linear. The speed up asymptotes around 30 CPUs where it is 6× faster than 1 CPU.*

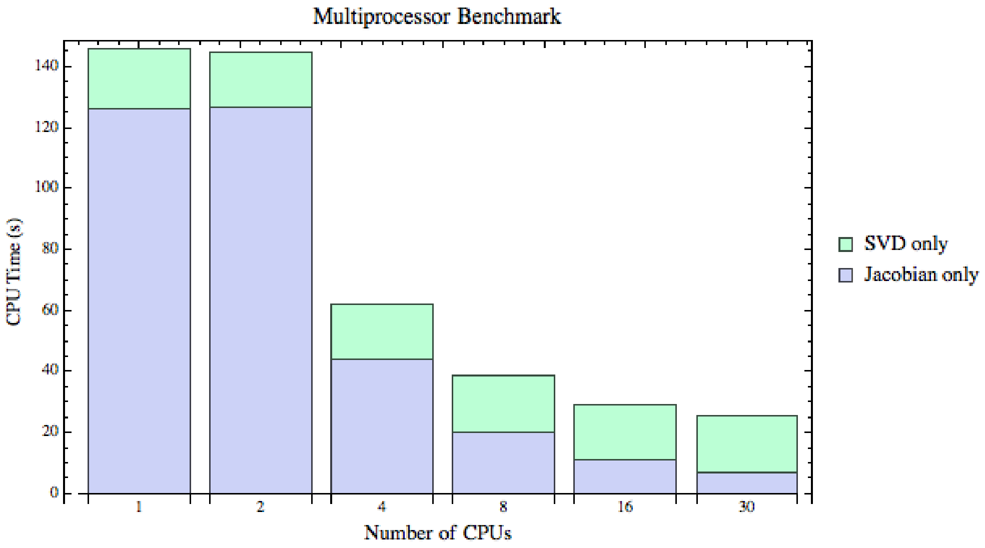

Figure 3: *The decrease in computation time comes from the calculation of the Jacobian because that is the only part that benefits from the increased number of CPUs. The SVD computation times remains constant because only 1 CPU is used.*

References

[1] M. McAteer et al, "Model Calibration and Optics Correction using Orbit Response Matrix in the Fermilab Booster", Proc. IPAC 2012, pg. 1251-1253, 2012.
[2] J. Safranek, "Experimental determination of Storage Ring Optics using Orbit Response Measurements", Nuclear Instruments & Methods A, Vol. 388, pg. 27-36, 1997.
[3] J. Safranek, "Linear Optics from Closed Orbits (LOCO) -- An Introduction", published in "Beam Dynamics Newsletter #44", edited by A. Ghodke & W. Chou, pg. 43-49, ICFA, 2007.
[4] J. DiMarco et al, "Test Results of the AC Field Measurements of Fermilab Booster Corrector Magnets", Proc. EPAC 2008, pg. 2347-2349, 2008.